プログラミング論I
(2)定数・変数，標準入出力

電子情報工学専攻　日浦 慎作

座席は自由です

C言語のプログラムの構造

```
#include <stdio.h>
int main(void)
{
    int seisu;
    seisu = 5;
    printf("seisuの値は%dです¥n",seisu);
    return 0;
}
```

変数（値を入れるための名前）の準備

変数定義が先

変数に値を入れる

値の表示

実行部分は後

それぞれの文は “;” で終わる（区切られる）.

C の文法

- 文 ・・・ ;（セミコロン）で終わる．処理実行の単位
  - seisu = 5;　　・・・ 変数 seisu に 5 を代入
  - printf(“result = %d¥n”, hensu);
    ・・・printf により画面表示する
- 識別子　・・・　変数，関数などの名前
  - 上の例では seisu, printf, hensu が該当
  - 自分で命名できる（すでに用意されているものもある）
- 文字列　・・・　“ と “ でくくられた文字
  - コンパイラは文字列を解釈しない．定数 5 などと同等
- 関数　　・・・　識別子(…) の形のもの

文法とプログラムの構造

```
#include <stdio.h>
int main(void) {
   int seisu;

   seisu = 5;
   printf("seisuの値は%dです¥n", seisu);
   return 0;
}
```

行頭が#はプリプロセッサ（特別扱い）

文字列
識別子
予約語
凡例

予約語一覧

auto	const	double	float	int	short	struct	unsigned
break	continue	else	for	long	signed	switch	void
case	default	enum	goto	register	sizeof	typedef	volatile
char	do	extern	if	return	static	union	while

識別子は，アルファベットか数字で作る．ただし↑の予約語は使えない
（ただし先頭はアルファベットのみ）

printf の使い方

バックスラッシュ ＼ は
¥ で入れる（¥ で表示されることも，＼で表示されることもある）

- 文字列を表示

  printf(“Hello World!”);

- 改行などの特殊文字も利用可能

  （¥n は改行を意味する）

  printf(“Hello World!¥n”);

- 数値を表示

  printf(“5 x 6 = %d¥n”, 30);

  - 出力結果は　5 x 6 = 30 となる（最後に改行）

  printf(“result = %d¥n”, hensu);

  - hensu = -7 のとき，出力結果は　result = -7

変数の種類

- 整数 (英語では integer)
  - 小数点以下は切り捨てられる
  - int (通常，32bit) が最も良く使われる
    - 他に char (8bit), long (32～64bit), short (16bit) など
- 実数（浮動小数点数）
  - 小数を含む計算に利用
  - double (64bit) が最も良く使われる
    - 他に float (32bit) , long double などもある

演算子について

- 加減乗除

  a + b　　a - b　　a * b　　a / b

- 剰余（割った余り）

  a % b　　a を b で割った余り（整数のみ）

- 優先順位

  数学と同じで * / のほうが + - よりも優先順位が高い

  ( ) を使うことができる（不安なときは使うべき）

- べき乗（a^b）の演算子はありません

代入について

- C言語の = は左側に書いた変数への「代入」
  - 右辺と左辺が後々まで等しいと決めるわけではない

    int a, b;

    a = 5;

    b = a;

    a = 3;

    とすると，bの値は5のまま（3にはならない）
  - 左辺には数式を書くことはできない

    int a;

    a + 5 = 7;

    とかいう書き方はできない（エラーになる）

定数について(1)

- 整数の代入

```
int a;
a = 5;
```

- 実数の代入

```
float a;
a = 5.23;
```

小数点があると，実数とみなされる

定数について(2)

- 整数の変数に代入すると

  int a;

  a = 5.23;　　小数点以下は切り捨てられる
  （5が代入される）

- 実数の変数に整数を入れることはもちろんOK

  float a;

  a = 5;　　普通に 5 が入る

ハマる人多数！要注意！

- 整数同士の計算は，整数になる

  float a;

  a = 3 / 2;　この場合，aの値は 1.5 ではなく 1 になる
- なぜか？
  - 3 や 2 は整数であり，整数同士の計算は整数
  - 実数の変数 a に代入されるときに初めて，実数に変換される
- 解決方法

  a = 3 / 2.0;　このように，どちらか一方に小数点（実数の定数）を使えば良い

printf と変数の型の関係

- 整数を表示するとき: %d を用いる

```
int a = 5;
printf(“値は %d¥n”, a);
```

- 実数を表示するとき: %f を用いる

```
float b = 5.23;
printf(“値は %f¥n”, b);
```

そのほかにもいろいろあります.

値の入力

- scanf を用いる
  - printf の反対の働きを持つと考えればいい.
    ただし一部注意が必要.
- 注意１：変数名の前に & をつける必要がある

  int a;

  scanf(“%d”, &a);
- 注意２：double の場合，%f でなくて %lf

  double b;

  scanf(“%lf”, &b);

  注意３：scanf では他の文章や ¥n はつけないのが安全

  注意４：scanf は連続して呼ばず，printf を間に挟むと安全

プログラム例：入力値の二乗を表示

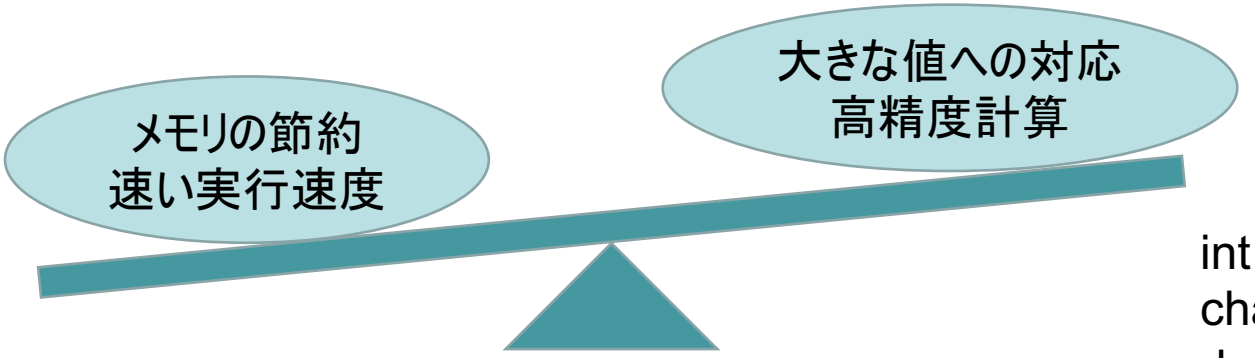
```
#include <stdio.h>
int main(void)
{
   int seisu;

   printf("整数を入力してください:");
   scanf("%d", &seisu);
   printf("その値の二乗は%dです¥n", seisu * seisu);
   return 0;
}
```

整数の変数の準備

値の入力. &を忘れないで！！

C言語の変数の型と入出力

変数の型	意味	データの大きさ	printf / scanf で表示・入力
int	普通の整数	32bit	%d
long	桁数の多い整数	32~64bit	%ld
char	桁数のとても小さい整数	8bit	%c（文字の入出力）
short	桁数の少ない整数	16bit	%hd
float	実数（浮動小数点数）	32bit	%f
double	倍精度実数	64bit	%lf



変数の型の選択は
プログラミングの要！

int ・・繰り返し処理の回数など
char ・・文字や画像の表現
double ・・科学技術計算など

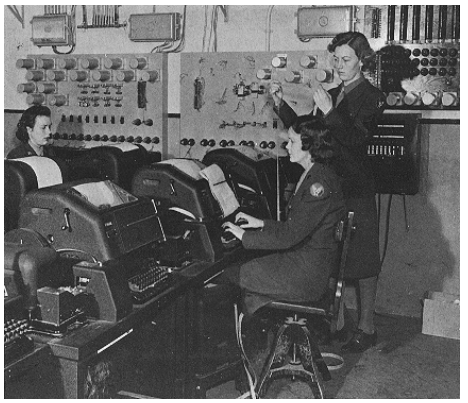
符号なし整数

- 整数型の変数については，
  符号付き（負の数を扱える）
  符号なし（負の数が扱えない）
  の2種類がある
- 普通は符号付きになるが，unsigned とつけると符号なしになる
- 扱える範囲がおおよそ倍になる

  int の場合 -2^{31}～$2^{31}-1$ の範囲

  unsigned int の場合 0～ $2^{32}-1$ の範囲

標準入出力とは



- 出力 ・・コマンドプロンプトへの文字表示
- 入力 ・・コマンドプロンプトでのキーボード入力
- 歴史的経緯（再掲）
  - コンピュータ登場前，遠隔地間の文書通信手段として，テレタイプという装置が使われていた
  - 第2次世界大戦後，コンピュータに「テレタイプ端末」を接続して文字の入力や結果の出力ができるようにした
    - 表示方式がプリンタから画面表示に変わっても，変わらずその文化が残り使い続けられている
- プログラム冒頭の #include <stdio.h> とは？
  - stdio = standard input / output = 標準入出力
  - #include は，「他のファイルを差し込む」ことを意味する
    - パソコン中の特別な場所に，printf や scanf について書かれたファイル stdio.h があり，それを #include <stdio.h> で読み込んでいる
    - 他に stdlib.h や math.h などもある（この授業の後半で講義）

プログラム例2：ふたつの数の和

```
#include <stdio.h>
int main(void)
{
  int a, b, c;

  printf("a =");
  scanf("%d", &a);
  printf("b =");
  scanf("%d", &b);
  c = a + b;
  printf("%d + %d = %d¥n", a, b, c);
  return 0;
}
```

プログラムのきれいな書き方

- 適宜スペースを置く
  - C言語で標準的なスペースの置き方は以下の通り
    - 演算子の前後にはスペースを置く　`c = a + b;`
    - , の後ろにはスペースを置く　`int a, b, c;`
  - 変数定義の後ろには1行，空行を置く
    - 適宜，さらに処理のまとまり事に空行をおいて良い
  - 段付け（インデント）について
    - `{` と `}` は複数のぶんをくくるのに使われる．
      `{` があると，次の行はタブ1つぶん下げる．
      `}` で閉じるときには，その行からタブ1つぶん上げる．
    - 次週以降の `if` や `for` を使うとさらに重要になる

コメント

- プログラム中に自由にかける「メモ」
  - コンパイラにより無視される（人間のためのもの）
- C言語のコメントの書き方
  - 方法1： /* で始まり */ で終わる部分はコメント
  - 方法2： // があると，その行のそれ以降はコメント

プログラム例2：ふたつの数の和

```
#include <stdio.h>
int main(void)
{
  int a, b, c;      // 変数の準備

  printf("a =");
  scanf("%d", &a);          /* a の入力 */
  printf("b =");
  scanf("%d", &b);          /* b の入力 */
  c = a + b;                /* 計算の実行 */
  printf("%d + %d = %d¥n", a, b, c);
  return 0;
}
```